

Mastering Software Robot Development Projects: Understanding the Association between System Attributes & Design Practices

Corinna Rutschi
Institute of Information Systems
University of Bern
corinna.rutschi@iwi.unibe.ch

Jens Dibbern
Institute of Information Systems
University of Bern
jens.dibbern@iwi.unibe.ch

Abstract

Software robots tend to increasingly take over organizational processes. However, little is known about principles of developing as opposed to using robotic systems, such as RPA robots and chatbots. Therefore, based on a comparative case study, this paper elaborates how different types of robots, due to distinguishing system attributes, relate to different design practices that arise from varying challenges of transforming existing routines into robots.

1. Introduction

Originally, humans executed organizational processes independently of machines, while nowadays, robotic systems often support or even substitute them [9]. This may involve a variety of different technologies including robotics systems such as robotic process automation (RPA) [24] and chatbots [17]. A robot can be any machine replacing work performed by humans [24] while gathering information and following instructions to execute tasks [22]. So far, it has been analyzed to what extent such robotic systems behave and interact with their respective environment [5], which kind of processes can be automated, and which factors make the introduction and use of such systems successful [24]. However, it has not yet been analyzed in detail to what extent the actual development of different robotic systems poses unique challenges and how these challenges can be adequately addressed. For this reason, this paper aims to answer the following research questions: *How do robotic systems differ from each other regarding different attributes? What are the challenges associated with such attributes and what practices can be applied to deal with them?* In order to answer these questions, two cases were analyzed in which on the one hand RPA robots and on the other hand a chatbot were

developed. Drawing on the conceptualization of system requirements or attributes by Demetis and Lee (2017), we identified the distinguishing attributes of robotic systems, such as their degree of autonomy. Based on a comparative case study we show that these attributes are associated with unique design practices that are grounded in challenges of developing new robotic-based routines. We use routine theory to theoretically explain the respective design practices.

2. Background literature & theoretical foundation

Robotic process automation. RPA enables the automation of business processes through the implementation of robots. Blue Prism, a software company, was the first to come up with the term RPA, allowing companies to automate business processes through technology, and more specifically through robots [20, 24]. RPA robots do not exist physically, but in the form of software systems [24]. They execute processes like humans while interacting with IT systems through their user interface [2, 24]. In doing so, RPA robots login (and out) of systems like humans do [24]. RPA thereby enables the human workforce to focus on more engaging and complex work [2, 14, 20]. After releasing an RPA robot into the live system, it performs business processes which do not require direct human interaction [21]. Clearly, in order to achieve the maximum outcome of RPA, companies need to learn how to manage RPA projects [24] and identify suitable business processes for RPA robot developments [24].

Chatbots. Another interesting approach of automating processes is to implement chatbots [17]. A chatbot represents a virtual assistant [19] that mimics human conversations. Thus, chatbots enable the automation of conversational processes [11]. They are made to interact and communicate with humans,

mostly in written form. Artificial intelligence (AI) hereby enables a chatbot to process natural language [17]. However, chatbots are subject to clear rules [10, 11]. The rules give the impression that the chatbot understands the human user, albeit the chatbot understands keywords and synonyms, and provides answers based on patterns [11]. In order to automate conversational processes, human experts need to structure conversations in decision trees that display every possible follow-up question and possible answers to those questions [10, 17]. After releasing a chatbot into the live system, the human user can interact with it via a user interface (UI), such as a pop-up window integrated on a website [17], Facebook Messenger, Skype or Slack [15]. Thereby, a chatbot is able to gather knowledge during each interaction with a human user and improve its accuracy of mapping incoming questions to correct answers within a corresponding decision tree [12, 17]. However, chatbots need training to improve their accuracy [17].

Similarities and differences of RPA robots and chatbots. The benefits of implementing RPA robots or chatbots range from decreased costs and error rates, improved process efficiency and customer satisfaction, and reachability of 24/7 [8, 10, 17, 18, 20]. Both, RPA and chatbot systems thereby allow the automation of certain types of organizational processes such as business processes (for RPA) [24] and conversational processes (for chatbots) [11]. Suitable processes for automation should be rule-based, non-complex, standardized and executed in high volumes [2, 8, 10, 17, 21, 24]. The development of both systems usually requires no in-depth programming knowledge, but rather processes can be graphically modeled within the respective system [15, 20, 24]. However, before processes can be automated through the development of the respective robots, appropriate process knowledge has to be acquired [10, 17, 21]. Although RPA robots and chatbots may be similar in some ways, they do show some differences such as in their degree of autonomy. While RPA systems enable rule-based automation, chatbot systems enable cognitive-based automation. Cognition hereby describes the ability to process numbers and text, to learn and to improve decision-making with an increasing amount of data [4]. Differences between both systems can be understood based on the system requirements or attributes of robots conceptualized by Demetis and Lee (2017), such as the *transformation process* (of inputs into outputs), *self-reference* and the *system/environment distinction*. Demetis and Lee (2017) focus their analysis on the usage of autonomous technologies such as robotic systems whereas in this paper the focus lays on their development. One system difference between

robots is that RPA robots transform structured input into structured output [21], whereas chatbots transform unstructured input into unstructured output, i.e. questions and answers in natural language [17]. This difference refers to the *transformation process*, i.e. to the transformation of an input into a technologized decision or output through “a complex nexus of technological interactions” [4, p.5751]. Human cognitive understanding is thereby replaced by technologized understanding which describes the acceptance or rejection of certain information, whereupon an algorithm-based computational response - a so-called technologized decision - is determined [5]. Another difference is that unlike RPA robots, chatbots can continuously improve their capabilities through training [12, 17] which reflects the system attribute *self-reference*. *Self-reference* refers to the process that helps a system collect information about itself, which in turn can help to change its way of functioning and to reproduce itself [5]. A further difference is that chatbots interact directly with humans [17], while RPA robots interact with other systems not requiring direct human interaction [2, 21, 24]. This reflects the system attribute *system/environment distinction*. Without its environment, no system can be perceived. It is important to understand what the environment of a system is and what relationships can be observed [5].

Challenges in robotic system developments in the light of routine theory. The specific attributes of different robotic systems pose challenges for the development of such systems. Essentially, challenges related to the replacement of existing processes - that are wholly or partly executed by humans - through a robotic system can be observed. This equals the challenge of developing a new routine where the artifact, i.e. the robot, takes over the role of the human actor. A routine can thereby be described as a series of interdependent actions performed on a pattern basis [7]. “Routines can be coded in cognitive artifacts such as work-flow graphs” [3, p.201], i.e. software systems, and consist of ostensive and performative aspects [3, 7, 16]. The ostensive aspect can be described as formal rules and procedures coded on the basis of organizational agencies’ experiences and learning. The performative aspect can be described as the execution of formal rules and procedures. The performative aspect is created by the ostensive aspect as well as vice versa [3, 7, 16]. This transformation of formal procedures into actual performance and vice versa requires *translation*. *Translation* describes the “co-production of formal procedures and performances” [3, p.205]. According to recent shifts in routine theory that put the artifact in the center, the routine has to be transformed into the artifact. In doing so, the artifact

partly takes over agency while becoming actor and influencing both the ostensive and the performative aspect of the routine. In the context of this paper, the artifact can be equated with the respective robotic system, i.e. the respective robot. Thus, for robots to execute routines, routines have to be transformed into the robot. Therefore, it is necessary to understand the ostensive and the performative aspect of the original routine as well as how the ostensive and the performative aspect need to look like when transformed into the robot. The robot works according to its own logic and thus requires routines to be structured accordingly, which might be different from the original structure of a human-executed routine. Therefore, to develop robots, it is necessary to decode routine knowledge in order to recode it as a robot. When the robot becomes the center of a routine, one no longer speaks of *translation*, but *inscription*. *Inscription* means to embed a range of rules and assumptions as “scripts” into artifacts, i.e. robots. It is not just about coding the original routine, but coding it in such a way that it can be executed by the robot. Thus, *inscription* can be defined as the designing, i.e. the development of robots consisting of *virtualization* and *actualization*. *Virtualization* refers to the translation of practitioners’ knowledge into formal rules and procedures, i.e. formal routines (ostensive aspect) or the *artifactual representations* of routines. *Artifactual representation* thus describes formal rules that have been transformed into an artifact, i.e. a robot. *Actualization* refers to the actual performance of formal routines (performative aspect) or *artifactual expressions* of routines. *Artifactual expression* describes the performance of such formal rules that have been transformed into an artifact, i.e. a robot. Artifacts, i.e. robots, are involved in co-creating knowledge and transforming actions, and thus also in the process of routine design [3]. Thus, RPA robots and chatbots may influence routines which can cause routines to change [16]. Two types of routines can be distinguished, namely dead and live routines. While dead routines are rigid and immutable, live routines are flexible and require the actor’s involvement and experience. Thus, when the robot becomes the center of the routine it influences and possibly changes the routine over time [3].

3. Methodological approach

In order to answer the research questions, a case study research method was chosen [6] and a multi-case design applied [25] where patterns related to similarities and differences between two cases were evaluated [23].

Data collection. Through theoretical sampling we identified two cases that seemed to contribute to answering the research questions. The cases consisted of two different robot projects realized at two different Swiss banks. For the sake of simplicity, they will be referred to as case 1 and case 2. We purposefully chose different cases in order to capture the differences in robotic systems in regard to Demetis & Lee’s (2017) design attributes. Between October and November 2017, we conducted several semi-structured interviews with people in different roles within the project teams in order to obtain a holistic picture [13, 25]. In addition, we analyzed further data such as robot software suit manuals.

Process of data analysis. After the interviews were conducted, they were transcribed. The qualitative analysis of the interview data was then accomplished in three steps. In a first step, we paired the initial and focused coding methods of Charmaz (2014) and applied them to the data to inductively identify important and relevant quotes out of the interview data that seemed useful for answering the research questions and understanding the cases. Subsequently, we applied the axial coding method of Strauss and Corbin (1990) to group the outcomes of initial and focused coding. During axial coding, categories or topics evolved that formed the basis of the case description. Initially, four major stages in the robot development emerged that were visible in both cases (see also the structure of the case description in chapter 4): *Build understanding of robot design; fit process and robot design; model processes; finalize development*. Subsequently, we deductively derived information related to the three above-introduced system requirements by Demetis and Lee (2017) from the data. The associated characteristics for RPA robots and chatbots are described in chapter 5. In a second step (selective coding), we inductively analyzed which challenges could be derived regarding the attributes and which design practices were used to address these challenges in each case. The associated practices are again shown in chapter 5. Finally, we used routine theory to theoretically explain the identified practices and their implications on routines, i.e. processes in chapter 6.

4. Case description

In both cases, the banks wanted to optimize its contact center (CC) in terms of efficiency. As a result, the banks wanted to improve performance and save costs. Different banks were thereby involved in case 1

and case 2. In case 1, RPA robots were introduced to automate business processes. The project was initiated in July 2017 and the first phase should be finalized in January 2018. In case 2, a chatbot was introduced to automate conversational processes. The project was initiated in October 2016 and the first phase should be finalized in December 2018. Overall, the software robots were successfully developed and implemented in both cases in conjunction with a certain degree of learning.

Case 1. Build understanding of robot design. Before the development of the RPA robots could be initiated in case 1, the project team had to understand the RPA robot design. This was critical, because it determined how business processes could be introduced to the RPA system so that an RPA robot could execute them. In case 1, Blue Prism's RPA system was used. Blue Prism allowed the programming of RPA robots that are capable of performing a sequence of process steps and mimicking what the human user normally does. The automation of business processes through the development of RPA robots was thereby done in Blue Prism Studio, which is divided into Process Studio and Object Studio. Process Studio enabled the configuration of the process logic and the business rules. Object Studio enabled the creation of reusable objects [20]. A process describes the logic of how a specific RPA robot executed tasks. An object describes the RPA robot's interaction with specific systems on their UI. The developers did not actually have to program the automation of business processes, but could graphically model them with the help of various flowchart elements. In Process Studio, one could either entirely model business processes or split them into multiple process steps. Each process step could be modeled in a separate page. Throughout all the pages, the main process could be kept slim on the main process page; frequently used process steps within a particular process could be reused. In Object Studio objects could be created, which allowed integrating external systems into the Blue Prism environment. With the 'spying mode' of Object Studio, every system button could be tracked and added to the corresponding object. Once a system and its entire corresponding buttons had been integrated, actions linked to the usage of a specific system could be modeled. Unlike in Process Studio, pages were hereby used to model individual actions related to a specific object. For example, in one of the business processes to be automated, the RPA robot had to send a confirmation letter to a customer who had opened a new account. For this purpose, the RPA robot had to know the respective system button "print" and execute the action "print confirmation letter". To then add an action to a

process in Process Studio, one could access the corresponding action from the Object Studio. To do this, the flowchart element "action" had to be inserted into the main process or a process step page in Process Studio. In summary, Object Studio enabled the integration of specific systems needed so that the RPA robots could execute the business processes modeled in Process Studio. Before the project team was able to identify suitable business processes for automation, it had to understand the RPA robot design described above. The developers had to clearly distinguish between processes and objects. As the project team was not yet experienced in RPA, it had to go through a learning curve. *"You gain experience on how the system works...at the beginning there is much difficulty before work. Thereafter it's just a circle."* (Supplier Chief Developer).

Fit process and robot design. The project team identified six criteria that determined whether a business process was suitable for an RPA-based automation or not. A business process had to be executed in (1) *high volume* and (2) *on a computer*, it had to be (3) *rule-based* and should (4) *entail limited exceptions*, it should (5) *implicate structured data* and (6) *each business process to be automated should replace 0.3 full-time equivalents (FTEs)* in order to achieve the break-even point after one year. The underlying assumption to reach the 0.3 FTEs was that the RPA robot's development costs were around CHF 60,000 and the costs of one FTE around CHF 200,000 per year. Thus, the development costs of one RPA robot equaled one third of the yearly costs of one FTE. Hence, it was only worthwhile to develop a robot in case it could undertake the work of 0.3 FTEs. Based on the criteria and a list of all processes executed in the CC, the project team identified nine business processes with automation potential. In a next step, these processes were analyzed in depth in order to ascertain whether they actually bring with them automation potential. During the in-depth analysis, it became clear that only four of the original nine business processes had real automation potential. Thus, those four processes should be automated in a first phase while potential additional processes should follow later. Once it had been determined that a business process had real automation potential, a process design document (PDD) and a solution design document (SDD) were created. The PDD described the current state of the process or the original process, and the SDD described the target state of the process and the basis for the RPA robot development. The SDD was necessary because not every process could be automated in its original form. Some processes had to be optimized and adapted according to the robot design, which was documented in the SDD. *"And then you see which parts of the*

process can be robotized and which cannot. And that is already the indication for the target process, i.e. for an SDD." (Supplier Project Manager 2). Thus, in order to automate a business process with the help of RPA robots, a detailed, explicit documentation of the respective business processes had to be created first. The necessary process knowledge was sometimes available explicitly and partially implicitly in the consciousness of the workforce. Once sufficient process knowledge was gathered and the PDD and SDD documents were created, the development of the RPA robots could be initiated.

Model processes. The development of the RPA robots was initiated with the modeling of the business processes defined in the SDD. This was done within Process and Object Studio. Each RPA robot was hereby set up through one process containing various objects that described the actions an RPA robot had to take in various process steps. However, not every developer understood this from the beginning, while some developers initially even created RPA robots within objects instead of forming processes by using objects. *"The object is something that you can re-use. The process is something you are only using for the current robotic process. So...you should not create a process inside an object. But many times they did it."* (Supplier Chief Developer). If done so, objects could only be used for one specific process, while reuse was not possible. However, the idea of using objects to build processes was to be able to reuse the objects for several processes involving the same systems. Even though this approach required more effort in the beginning, it allowed a faster development of subsequent RPA robots accessing the same systems. *"Because the first robots are always the hardest. How so? Because...you develop that in objects. These are objects that can be reused in other robots. This automatically means that subsequent robots can be developed faster."* (Supplier Project Manager 2). Once the chief developer discovered that the other developers defined processes within objects instead of using various objects to define one process, he drew their attention to it and they changed their approach from object-based to process-based development.

Finalize development. Once an RPA robot was developed and its performance was tested. An RPA robot passed the testing if it was able to complete its business process without errors. If an error occurred, the developers had to fix it before the robot could be re-tested. Once an RPA robot had finally passed the testing, it was implemented into the live system. Thereafter, it ran independently. After a period of five months, the first RPA robot was released on 20th November 2017. Once an RPA robot was implemented in the live system, no further expansion of its

capabilities were added unless environmental changes occurred.

Case 2. Build understanding of robot design. Before the development of the chatbot could be initiated in case 2, the project team had to understand the chatbot design. This was critical, because it determined how conversational processes could be introduced to the chatbot system so that the chatbot could execute them. In case 2, Nuance's chatbot system Nina was used. Nina is a virtual assistant or chatbot who can understand natural language and improve its performance over time with the help of human interactions [1]. Nuance offered various tools enabling the development of Nina, i.e. the automation of conversational processes. The developers did not actually have to program the automation of conversational processes, but could graphically model them. Nuance IQ Studio enabled the modeling of conversations in decision trees directly within the chatbot system. In addition to modeling decision trees, variations of questions and synonyms also had to be implemented so that the chatbot could ultimately interact with the human end user as smoothly as possible. Nuance Experience Studio was therefore used to implement grammar, variations and synonyms, so that the chatbot could understand the language of the end users. Nuance Analytics enabled the monitoring of the chatbot and its usage, and the review of end user chats. Nuance Software Developer Kit enabled the implementation of the chatbot on the bank's website and the storage of end user chats on the cloud. Decision trees were modeled around one main question, which constituted the root, while possible direct answers and follow-up questions formed the branches of a decision tree. As an example, an end user might ask "How can I open a new bank account?" upon which the chatbot might ask back "Are you a private or a business client?". Each decision tree should preferably model all possible conversations around one specific main question. *"So first you have the main questions defining entry points if you like. Then you had to define the answers. Thus, one or x answers fit to one main question. And then you can also have one or x questions that map to this main question."* (Supplier Project Manager). Before the project team was able to identify suitable conversational processes for automation respectively suitable main questions, it had to understand the robot design described above. As the project team was not very experienced with chatbots yet, it had to go through a learning curve. *"A very new topic. Is it, I believe, in every company."* (Client Project Manager).

Fit process and robot design. Not only the chatbot design determined the structure of conversational

processes, but also the end users influenced how conversational processes took place. The project team had to understand the end users and how they would ask questions, to then efficiently model conversational processes according to the chatbot design. *"If that thing [the chatbot] does not provide the answers the user needs, then he [the user] will not use it."* (Supplier Project Manager). The project team identified five criteria that determined whether a main question was suitable for a chatbot-based automation or not. A main question had to (1) *contain general information*, (2) *allow an easy modelling of a conversation around it*, (3) *occur in high volume*, (4) *contain self-service components* or aspects the end user could do himself and (5) *be related to a non-value added process*. Up until then, client 2 was tracking every incoming customer question in a customer relationship management (CRM) system. Thus, process data was already available. This served as a starting point to identify suitable main questions around which conversations could be modeled. Based on the criteria, the project team identified ten main questions with automation potential that should be implemented first.

Model of processes. Before the automation could be initiated, the project team had to define a content strategy determining the behavior of the chatbot in terms of how the chatbot should act if it did not understand a question, say goodbye, end a conversation, connect an end user to a call agent or direct an end user to the self-service. The content strategy and some social questions that had already been incorporated into Nuance's chatbot system formed the backbone of the chatbot. The development of the chatbot was then initiated with the modeling of conversational processes around the ten selected main questions and the implementation of variations and synonyms within the chatbot system. One main question required about 100 variations, so that the chatbot was able to answer accurately. *"Still, if there is a 101st question and the syntax is wrong, we are pretty sure the chatbot is going to map the question to the right main question."* (Supplier Project Manager). In order to optimally model the decision trees, the project team resorted to the implicit knowledge of 150 call agents. *"They [the call agents] are in constant contact with the end user and know how the end user is ticking."* (Client Project Manager). The 150 call agents supported the project team in modeling the decision trees and implementing variations and synonyms around the ten initially selected main questions and later around additional questions. However, conversational processes not only had to be modeled and variations and synonyms implemented, but the chatbot also needed training to continually improve its accuracy. The 150 call agents again assisted the project

team by having conversations with the chatbot to test how it responded and thereby to train it. They tried to formulate the same questions as differently as possible to see if the chatbot still understood them. *"Then we look in the background, whether it worked or not, and if not we occasionally pull certain connections manually, if the chatbot makes a wrong matching. But the front agent always confirms whether the right or the wrong answer has arrived. The agent enters a variation and the chatbot then asks 'are you satisfied with my answer?' and then he [the agent] says yes or no and then he [the chatbot] learns these variations."* (Client Project Manager). Over time, this helped the chatbot to correctly answer questions that aimed for the same answer but were worded differently. Any questions the chatbot could not answer were collected with the help of Nuance Analytics and could be implemented by the project team as an extension of existing decision trees, to model new decision trees or as variations or synonyms.

Finalize development. After a period of eleven months, the chatbot was released to the live system on 23rd August 2017. Subsequently, the employees and the end users were able to access the chatbot. For the time being, however, the release was only announced internally. From this point onwards, not only the 150 call agents could train the chatbot, but the internal workforce was also asked to train the chatbot. Again, unanswered questions could be implemented as an extension or to model new decision trees, or as variations or synonyms. After another three months of expanding decision trees and implementing new decision trees, and variants and synonyms, the chatbot was announced externally on 28th November 2017. From then on, the end users could use the chatbot. They trained the chatbot indirectly and unanswered questions could still be implemented continuously. Thus, even after the implementation of the chatbot into the live system, expansion of its capabilities could be implemented, continuously, enabled through on-going training.

5. Cross-case analysis

Based on our deductive application of the attributes *transformation process*, *self-reference* and *system/environment distinction* to the data, we identified a sub-attribute of the *transformation process* attribute, i.e. *autonomy of technologized decision-making*. This refers to how far a robot is capable of making decisions on its own. Overall, the characteristics of the attributes differ in both cases. In regard to Demetis and Lee (2017) one could say that the RPA robots perform technologized decision-

making by transforming structured input into structured output while the chatbot does so by transforming unstructured input into unstructured output (*transformation process*). Structured input refers to data retrieved from systems that case 1's RPA robots could access through respective objects defined in Object Studio. Unstructured input and output refers to questions in natural language asked by the human users and answered by case 2's chatbot. Both robotic systems are thereby based on clear rules, which limited the variation of successive process steps in case 1 but not directly in case 2. In case 1, the RPA robots should execute processes exactly according to given rules, while in case 2 processes could change due to human-chatbot interactions. Thus, one could say that case 1's RPA robots are less autonomous than case 2's chatbot (*autonomy of technologized decision-making*). In addition, case 2's chatbot is able to learn and improve its accuracy through training and referring to itself, which does not work for case 1's RPA robots (*self-reference*). Referring to itself or self-referential hereby means that based on subsequent inputs or questions from human users, the chatbot is able to judge whether its outputs or answers were appropriate. Finally, in order for case 1's RPA robots to perform the appropriate business processes, they needed to interact with other systems and retrieve certain data from those systems. For case 2's chatbot to be able to perform the appropriate conversational processes, it needed to interact with human users. Thus, in case 1 other systems whereas in case 2 humans are in the environment of the robotic system (*system/environment distinction*). The different characteristics of the attributes relate to challenges, such as that processes have to be automated depending on the robot design (and the human user), that chatbots can indeed learn, but have to be trained for it and that RPA robots can execute processes only if they can interact with other systems. After applying the attributes of Demetis and Lee (2017) to the interview data, we examined whether design practices, dealing with challenges associated with different characteristics of the above-mentioned attributes, could be derived from the data. In summary, partly different and partly similar design practices could be identified. In both cases, an *understanding of the robot design* and of how it was composed had to be gained first, before the respective robots could be developed efficiently. Case 1's developers had to clearly distinguish between processes and objects. Case 2's developers needed to gain an understanding of how decision trees could be modeled, and variations and synonyms implemented. This helped *defining process selection criteria* that again enabled *identifying appropriate processes*. Case 1's project team then had to gain an *understanding of the identified processes*

and document their current state. However, the structure of the identified processes did not always conform to the specifications of the RPA robot design. For this reason, certain processes had to be adapted and newly *documented as target processes* according to the robot design. This was not necessary in case 2, but the project team had to *define a content strategy* guiding the chatbot's behavior. Subsequently, the development of the respective robots could be initiated in both cases by *modeling processes and object* in case 1, and *modeling decision trees and implementing synonyms and variations* in case 2. Thus, five practices could be identified that relate to the attribute *transformation process* that partially differ for RPA respectively chatbot developments: *understanding the robot design*; *defining process selection criteria*; *identifying appropriate processes*; *document the current and the target state of the identified processes respectively define a content strategy*; and *modeling processes and objects respectively modeling decision trees and implementing synonyms and variations*. Regarding the sub-attribute *autonomy of technologized decision-making*, case 1's RPA robots strictly followed given rules, while case 2's chatbot had variability in how to conform to given rules depending on unpredictable behavior of the human users (e.g. how a particular question is asked). After completing the development of a respective robot, the project teams of both cases had to *test the robot* and make sure that it performed the respective processes faultlessly. The testing in case 2 not only helped to detect mistakes, but also to *train the chatbot*. This enabled the chatbot to learn and improve its accuracy. Thus, in case 1, one cannot speak of *self-reference* in principle, since the RPA robots were not able to learn, to improve their accuracy, or to refer to themselves, which was applicable in case 2. Finally, to account for the attribute *system/environment distinction*, it can be observed that in case 1 other systems are in the environment of the RPA system, while in case 2 humans are in the environment of the chatbot system. For case 1's RPA robots to be able to interact with these systems, the *systems had to be integrated* via creating objects in Object Studio. In order for case 2's chatbot to interact smoothly with human users, the project team needed to gain an *understanding of the human user behavior* to appropriately model conversational processes. Only then the human user would use the chatbot.

6. Discussion

The analysis of the two cases showed that different robotic systems due to differences in system attributes relate to different design practices which are grounded

in respective challenges of developing software robots. Essentially, these challenges can be summarized as the development of a new routine performed by a robot. Accordingly, routine theory was informative to understand the implications of varying design practices observed for the development of different types of robots. The processes performed by robotic systems can be compared to routines. Previously it has been assumed that humans, but not artifacts have a direct influence on routines, however recent research brings the artifact to the center of routines [3]. Figure 1 illustrates how key concepts of routine theory relate to the identified design practices from the case analysis. The design practices are numbered whereby all numbers followed by an “a” refer to chatbot development and those followed by a “b” to RPA robot development design practices. Figure 1 also illustrates which of the design practices were used to deal with which of the attributes *transformation process*, *self-reference* and *system/environment distinction*.

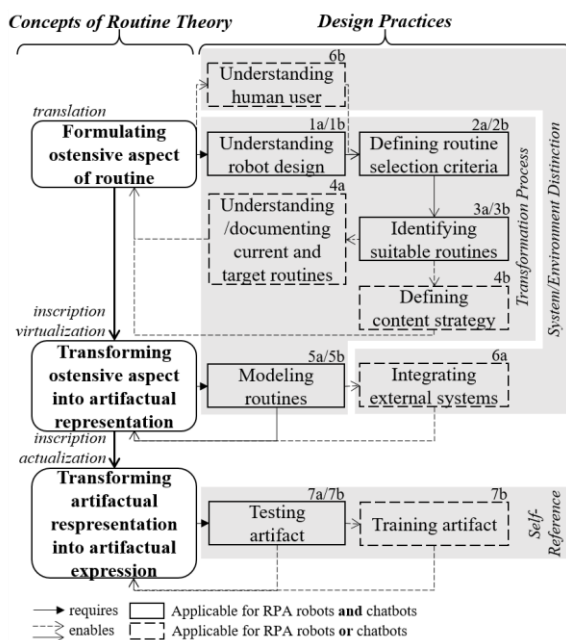


Figure 1. Design practices related to routine theory

In order for robots to execute an existing routine, the routine has to be transformed into a robotic routine, i.e. a new routine performed by a software robot. However, as far as the robot follows its own unique logic of executing a process, this logic has to be learned by those developing the robot. In other words robots influence the development of the new routine in that an *understanding of the robot design (1a/1b)* has to be gained first, in order to thereafter formulate routines appropriately (so that they fit the robot). The skills and capabilities of the actors, i.e. the developers

are thereby conveyed and transformed by the capabilities of the robot they seek to design [3]. By analyzing the cases it could be seen that the robot design significantly determined what types of processes, i.e. routines could be automated in which form. The development of RPA robots required business processes to be modeled in processes and objects, whereas the development of a chatbot required conversational processes to be modeled in decision trees and a separate implementation of variations and synonyms. The primary purpose of objects aiming to integrate external systems is to be able to reuse them for different processes. Thereby, RPA robots are subject to limited and apriori known processes. A decision tree defines the general logic of a chatbot, but it can be continually expanded and variations and synonyms help to improve the chatbot's understanding of humans. Thus, an *understanding of the human user (6b)* behavior had to be gained in terms of efficiently develop a chatbot. This was followed by the *definition of process, i.e. routine selection criteria (2a/2b)* that determined the selection of suitable processes in regard to the respective robot design. This then allowed the *identification of suitable processes, i.e. routines (3a/3b)* for both the RPA and chatbot development. Subsequently, a detailed *understanding and an explicit documentation of each process (4a)* to be automated by RPA robots had to be elaborated. The explicit process, i.e. routine documentation serves as the basis for the development of corresponding RPA robots and can be associated with the *formulation of the ostensive aspect of the routines* or the *translation* of formal rules and procedures into routines. Before modeling conversational processes a *content strategy needs to be defined (4b)* that specifies the chatbot's behavior. Subsequently, processes, i.e. routines could be modeled directly in the chatbot system, without resorting to an explicit process documentation. D'Addario (2011) describes this transformation of routines into artifacts, i.e. robots, as *inscription*. *Inscription* thereby enables the delegation of so far human-owned processes, i.e. routines, to robots [3]. In order to transfer this into the context of this paper, one could describe *inscription* as the actual development or building of the respective robots. The development of robots thereby basically means to transform certain processes into robots. *Inscription* consists of *virtualization* and *actualization* [3]. In both cases, *virtualization* can be related to the *modeling of initially identified routines*, i.e. processes or conversations around main questions, within each robotic system according to the respective robot design. By modeling processes and objects in case of an RPA robot development and decision trees in case of a chatbot development, processes, i.e. routines could be

transformed into the respective robots. In order for RPA robots to perform routines they had to interact with other systems. *Integrating external systems (6a)* into the robotic system through the creation of objects was therefore required. The modeling of processes can be linked to the formulation of rules that guide the behavior of a particular robot. As already introduced earlier, the ostensive aspect of a routine could be described as corresponding formal rules and procedures that make up the routine. Thus, rules inscribed to a respective robot can be considered as the ostensive aspect or the *artifactual representation* of a specific routine. The ostensive aspect of a routine is not only the basis for the performative aspect of the same routine but is simultaneously influenced by the performative aspect [3]. Thereby, the *artifactual representation* (ostensive aspect inscribed into the artifact) may imperfectly represent the *artifactual expression* (performative aspect inscribed into the artifact). Once rules are incorporated into artifacts, i.e. robots, they can become more stable. However, routines cannot always be perfectly transformed into robots, while *artifactual representations* may not always perfectly imitate actual routines, i.e. *artifactual expressions* [3]. Therefore, *testing (7a/7b)* is required to check for both chatbots and RPA robots whether they perform their respective processes without failure. If errors can be identified, they have to be corrected. Thus, through *testing* it was examined whether the *artifactual representation* coincided with the *artifactual expression* of a respective routine. RPA robots clearly influenced the *artifactual representation* through their robot design, however, they had no influence on the *artifactual expression*, as this clearly depended on the ostensive aspect (the *artifactual representation*). A chatbot does not solely require *testing* but *training (7b)* simultaneously. *Training* helped the chatbot to learn and thus influence the *artifactual expression*, which in turn influenced the *artifactual representation*. Thus, the chatbot influenced the *artifactual representation* through its robot design, as well as the *artifactual expression*, since the chatbot could learn and improve its accuracy. Routine theory distinguishes different types of routines such as live and dead routines. Dead routines are rather rigid, while live routines are flexible and can be changed by their actors [3]. In this context, the actor could be associated with the artifact, i.e. the robot and in the case of a chatbot additionally with the human user. One could say that RPA robots follow dead routines while chatbots follow live routines. Although an RPA robot's design initially influences the ostensive aspect of the routine, once the ostensive aspect has been implemented into an RPA robot, it does not change, unless errors occur during the transformation from

artifactual representation to artifactual expression. A chatbot has an initial and later influence on the ostensive and the performative aspect of a routine and can influence *artifactual representation* as well as *artifactual expression* during the transformation from one into the other and vice versa.

7. Conclusion, limitations & future research

In conclusion, it can be said that different robotic systems exhibit unique characteristics along a set of system attributes. These characteristics relate to challenges regarding the development of software robots, i.e. the development of new routines performed by a robot. Through our analysis, we have identified a variety of design practices that help address these challenges. Our research extends previous research that has focused on analyzing differences of robotic systems with regard to the usage of such systems, but not with regard to the actual development of software robots. We show that differences can lead to unique challenges related to the robot development, i.e. the transformation of an existing routine into a robot. The artifact, i.e. the robot does hereby no longer simply fulfil a supporting function, but also takes on agency while influencing routines [7]. The fact that humans still identify and select the routines to be performed, and thus determine the capabilities of robots, is nothing new. What is new, however, is that the routines to be automated have to be adapted to the robot. Thereby, different robots influence routines diversely. While the design of a respective robot has an impact on the ostensive aspect of a routine, the performative aspect of the routine may be affected, as long as the routine involves uncontrollable external actors, such as for example human users interacting with a chatbot. Thus, the extent to which routines have to be adjusted to the robot depends on the characteristics of certain attributes, which in turn can lead to challenges that can be addressed using different design practices. We are aware of the fact that our results are limited to two cases regarding two different robotic systems. Therefore, we aim to further extend our data sample in a next step to verify and extend our model. Beyond that, our research paves the way for future research into the efficient implementation and development of robotic systems. For example, future research could delve deeper into opening the black box of robot design logic and how humans can understand and translate routines to robots. Specifically, as robots become ever smarter through the use of AI.

8. References

- [1] Intelligent Self-Service with Nina, Nuance Communications, 2018, [Online] Available from: https://www.nuance.com/content/dam/nuance/en_us/collateral/enterprise/data-sheet/ds-nuance-nina-v2-en-us.pdf, [Accessed 14 May 2018].
- [2] Asatiani, A. and E. Penttinen, Turning Robotic Process Automation into Commercial Success—Case Opuscapita, *Journal of Information Technology Teaching Cases*, 2016, 6(2): pp. 67-74.
- [3] D'Adderio, L., Artifacts at the Centre of Routines: Performing the Material Turn in Routines Theory, *Journal of Institutional Economics*, 2011, 7(2): pp. 197-230.
- [4] Davenport, T.H. and J. Kirby, Just How Smart Are Smart Machines?, *MIT Sloan Management Review*, 2016, 57(3): pp. 21-25.
- [5] Demetis, D. and A. Lee, When Humans Using the It Artifact Becomes It Using the Human Artifact, 2017.
- [6] Eisenhardt, K.M., Building Theories from Case Study Research, *Academy of management review*, 1989, 14(4): pp. 532-550.
- [7] Feldman, M.S., B.T. Pentland, L. D'Adderio, and N. Lazaric, Beyond Routines as Things: Introduction to the Special Issue on Routine Dynamics, *Organization Science*, 2016, 27(3): pp. 505-513.
- [8] Fung, H.P., Criteria, Use Cases and Effects of Information Technology Process Automation (Itpa), 2014.
- [9] Georgakopoulos, D., M. Hornick, and A. Sheth, An Overview of Workflow Management: From Process Modeling to Workflow Automation Infrastructure, *Distributed and parallel Databases*, 1995, 3(2): pp. 119-153.
- [10] Guzman, I. and A. Pathania, Chatbots in Customer Service, Accenture, 2016, [Online] Available from: https://www.accenture.com/t00010101T000000_w_/br-pt/acnmedia/PDF45/Accenture-Chatbots-Customer-Service.pdf [Accessed 25 Oct 2017].
- [11] Heller, B., M. Proctor, D. Mah, L. Jewell, and B. Cheung, Freudbot: An Investigation of Chatbot Technology in Distance Education, in *EdMedia: World Conference on Educational Media and Technology*, Association for the Advancement of Computing in Education (AACE), 2005.
- [12] Hildebrand, M., A. Eliëns, Z. Huang, and C. Visser, Interactive Agents Learning Their Environment, in *International Workshop on Intelligent Virtual Agents*, Springer, 2003.
- [13] Klein, H.K. and M.D. Myers, A Set of Principles for Conducting and Evaluating Interpretive Field Studies in Information Systems, *MIS quarterly*, 1999: pp. 67-93.
- [14] Lacity, M.C. and L.P. Willcocks, A New Approach to Automating Services, *MIT Sloan Management Review*, 2016.
- [15] Patil, A., K. Marimuthu, and R. Niranchana, Comparative Study of Cloud Platforms to Develop a Chatbot, *International Journal of Engineering & Technology*, 2017, 6(3): pp. 57-61.
- [16] Pentland, B.T. and M.S. Feldman, Organizational Routines as a Unit of Analysis, *Industrial and corporate change*, 2005, 14(5): pp. 793-815.
- [17] Sengupta, R. and S. Lakshman, Conversational Chatbots – Let's Chat, Deloitte, 2017, [Online] Available from: <https://www2.deloitte.com/content/dam/Deloitte/in/Documents/strategy/instrategy-innovation-conversational-chatbots-lets-chat-final-report-noexp.pdf> [Accessed 25 Oct 2017].
- [18] Sharma, P., R. Southern, and D. Dalton, The Disruptive Chat Bots - Sizing up Real Opportunities for Business, Deloitte, 2016, [Online] Available from: <https://www2.deloitte.com/content/dam/Deloitte/ie/Documents/ie-disruptivechat-bots.pdf> [Accessed 25 Oct 2017].
- [19] Shawar, B.A. and E. Atwell, Different Measurements Metrics to Evaluate a Chatbot System, in *Proceedings of the workshop on bridging the gap: Academic and industrial research in dialog technologies*, Association for Computational Linguistics, 2007.
- [20] Slaby, J.R., Robotic Automation Emerges as a Threat to Traditional Low-Cost Outsourcing, Hfs Research Ltd, 2012.
- [21] Sutherland, C., Framing a Constitution for Robotistan, Hfs Research, 2013.
- [22] Tirgul, C.S. and M.R. Naik, Artificial Intelligence and Robotics, *International Journal of Advanced Research in Computer Engineering & Technology*, 2016, 5(6): pp. 1787-1793.
- [23] Webster, J., Desktop Videoconferencing: Experiences of Complete Users, Wary Users, and Non-Users, *MIS quarterly*, 1998, 22(3): pp. 257-286.
- [24] Willcocks, L. and M.C. Lacity, *Service Automation: Robots and the Future of Work* Steve Brookes Publishing, 2016.
- [25] Yin, R.K., Case Study Research, in *Applied Social Research Methods Series*, 5, CA: Sage Publications, Beverly Hills, 1984.